LCFI Vertex Package

Presented at the 17th International Workshop on Vertex detectors
July 28 - 1 August 2008, Utö Island, Sweden

Sonja Hillert
on behalf of the LCFI Collaboration *

University of Oxford - Department of Physics
Denys Wilkinson Building, Keble Road, Oxford OX1 3RH - UK
E-mail: s.hillert1@physics.ox.ac.uk

Abstract

The LCFIVertex software, developed by the Linear Collider Flavour Identification (LCFI) collaboration and providing tools for vertexing, flavour tagging and quark charge determination for low-mass vertex detectors of high point resolution is presented. Particular emphasis is given to code extensions since the first release in April 2007. A recently developed new vertex finder, ZVMST, and its performance at $\sqrt{s} = 91.2\,\mathrm{GeV}$, are described in more detail.

1 Introduction

The International Linear Collider (ILC), a 200 - 500 GeV e^+e^- collider, is envisaged by the particle physics community as next major accelerator following the LHC [1, 2]. The high precision measurements planned at this machine place demanding constraints on the quality of the detectors as well as requiring excellent software for event reconstruction and data analysis. In the context of vertex detector R&D for the ILC, and more generally applicable for low-mass vertex detectors of high point resolution, software for vertexing, flavour tagging and quark charge determination has been developed. This code named LCFIVertex [3, 4] and interfaced to one of the ILC software frameworks, is currently widely used by both the ILD [5] and SiD [6] detector concept groups for detector optimisation and preparation of Letters of Intent due in spring 2009.

This paper is organised as follows: section 2 gives an overview of the framework to which the code is interfaced and a typical detector model used in the examples of code performance. An overview of core functionality of the code and of updates since the first release is given in section 3. One of the recent extensions of the package, the minimum spanning tree-based vertex finder ZVMST [7], is described in more detail in section 4. It is followed by a preliminary performance comparison between ZVMST and the leading vertex finder `ZVTOP_ZVRES` in section 5. Section 6 summarises the main results.

2 Marlin framework and vertex detector used in performance studies

The LCFIVertex code is based on the event data model LCIO [8] permitting the exchange of Monte Carlo (MC) samples between the different software frameworks in use by the ILC community, and the modular analysis framework Marlin [9], facilitating distributed code development.

The MC program Pythia was used to generate events, which were passed through the GEANT4-based [10] detector simulation MOKKA [11] using the `LDC01_05Sc` detector model [12]. Events were reconstructed using the MarlinReco package [13], in particular the digitization and `FullLDCTracking` code by A. Raspereza [14] and the particle flow package PandoraPFA by M. Thomson [15].

In the detector model `LDC01_05Sc`, developed by the LDC detector concept group, now part of the ILD group, the vertex detector design envisages 5 layers of sensors with radial positions, numbers of sensors in the layers and sensor dimensions as described in the TESLA TDR [1], i.e. five evenly spaced layers with the innermost radius being 15 mm and that of the outermost layer being 60 mm. Sensors and support together are assumed to correspond to a material budget of $0.1\% X_0$ per layer. The point resolution used in the MC test sample is $2.8\,\mu m$, as currently used in the benchmark studies performed by the ILD detector concept group. Since the forward region of the detector is of particular importance at the ILC, and hermeticity of all systems is hence crucial, vertex detector R&D groups aim at a polar angle coverage of $|\cos\theta| \leq 0.96$. The B-field is assumed to be 4 T.

3 Overview of the LCFIVertex software

The first version of the LCFIVertex code was released in April 2007 [16]. In developing this new C++ based code the LCFI physics group has been building on earlier work by the SLD collaboration [17, 18] and by LCFI and the TESLA detector R&D group [19, 20, 21, 1], permitting detailed code validation using the fast MC program SGV [22] for which an interface to a FORTRAN implementation of the core algorithms was available [23].

The core modules of LCFIVertex, all run on a jet-by-jet basis, provide the vertex finder `ZVTOP`, a neural-net based flavour tag following the method developed by R. Hawkings and quark charge reconstruction. The `ZVTOP` vertex finder comprises three algorithms, each based on a different ansatz for finding decay vertices of heavy flavour hadrons from the topological information contained in the tracks of the input jet. The leading algorithm most widely used in the ILC community is the `ZVRES` approach; its name refers to the fact that the algorithm strongly relies on a criterion for checking whether different vertices are resolved from each other. In contrast, the `ZVKIN` or ghost track algorithm is a more specialised approach dedicated to b-jets, which approximates the direction of flight of the decayed B-hadron and uses this additional kinematic information to identify the decay vertices. This approach permits vertices to be found also in jets, in which both the B- and the subsequent charm decay are one-pronged. The new ZVMST approach resembles `ZVRES` in scope and mathematical description of the topological information but uses a minimum spanning tree (MST) to select the most promising vertex candidates.

Observables based on secondary vertices, such as the P_T-corrected vertex mass [24], the secondary vertex probability and its decay length, provide the most stringent indication of the jet flavour. Such variables therefore form the most important input to the flavour tagging neural networks used for jets with at least two reconstructed vertices. If only the primary vertex is found, e.g. because the heavy flavour decay occurs so close to it that the two vertices cannot be resolved from each other, different variables are used to distinguish between jet flavours. In that case, the best variable identified so far is the joint probability of all tracks to originate from the primary vertex, which is calculated independently from the impact parameter significances in the R-ϕ plane and in z. It is complemented by impact parameter and impact parameter significance as well as momentum of the two most significant tracks in the jet.

Altogether, the LCFIVertex code provides 9 neural networks, three each for the three cases that exactly one, exactly two and at least three vertices are found by `ZVTOP`. Of the three networks for each of the cases, one is used to identify b-jets, one to identify c-jets in a sample with arbitrary background composition including all jet flavours and one to identify c-jets for samples for which the background is known to consist of b-jets only (this is the case for some physics processes, and permits better c-jet identification compared to the inclusive tag). A more detailed description of the different inputs and the relative weight of their contribution to the flavour tag is given elsewhere [19, 3]. While this method forms the default approach, ensuring a high degree of flexibility has been a key aspect in developing the new C++ code; so network architecture, node type and training algorithm as well as input variables can be modified and improved or adjusted by the user to meet the special needs of a given analysis.

The quark charge algorithm builds strongly on the successful SLD method [18], with modifications developed subsequently by LCFI [25]. With the C++ implementation of the `ZVKIN` algorithm [26], the foundation for use of the SLD charge dipole method in the ILC context has been laid; however, further studies will be required e.g. to optimise `ZVKIN` parameters.

Since the first release that made available these core algorithms, the package has

been considerably extended and improved in performance. A module for identification and removal of tracks from K_S and Λ decays and from photon conversions was added (K. Harder), and code implemented to perform a fit of the impact parameter significance distribution (E. Devetak), required for calculation of one of the flavour tag inputs. Both are described in more detail elsewhere [3].

Technical improvements include the move of vertex charge calculation to a dedicated processor to decouple its implementation from the calculation of the flavour tag inputs (E. Devetak), an interface to a Kalman filter by Gorbunov and Kisel (T. Lastovicka), leading to increased speed of the IP fit processor, and code to ensure the package is compatible with the DST format recently agreed by the ILD concept group (C. Lynch).

Focus of current work is the optimisation of track selection and other code parameters as well as the training of new neural networks based on GEANT4-MC and full reconstruction (R. Walsh), presented at recent meetings and conferences, see e.g. [27]. This work is aimed at providing a tuned code configuration for ILD detector optimisation towards the LoI.

Also in view of the benchmark studies for the LoI, extensive diagnostic tools have been added (V. Martin), including plots of all flavour tag inputs and neural network output variables, each both inclusively and separately for the 1-, 2- and 3-vertex category, tables of efficiency, purity and corresponding neural network cut value and graphs of purity vs. efficiency as well as of mistag rates for all tags.

In terms of new algorithms, a "vertex cheater" using MC information on which tracks originate at the same space point and the new `ZVMST` vertex finding approach have recently been developed (S. Hillert).

4 The new vertex finder algorithm ZVMST

Minimum spanning trees (MSTs) are a mathematical optimisation tool with a wide range of applications such as in source detection in gamma ray images [28], where this approach dates back to the early 1980s. As the method exploits topological information - in the astrophysical example the connectedness of the detected photons - it provides a natural approach to topological vertex finding. Mathematically, minimum spanning trees are a special type of graph. A graph is a set of nodes connected by edges that can have weights assigned to them. Trees are graphs that do not contain any loops. For graphs with weighted edges, the minimum spanning tree is defined as the tree that minimizes the overall weight in the subset of graph edges selected. It has been mathematically proven that if no two weights are equal there always exists a unique solution for the minimum spanning tree. Efficient algorithms for finding this solution exist and optimized implementations are available in the graph library of the C++ package boost [29], already used by the LCFIVertex package.

A central idea of the standard `ZVTOP_ZVRES` algorithm is to describe each track by a probability density function $f_i(\vec{r})$ in 3D space and to use these to define a vertex function $V(\vec{r})$ that yields higher values in the vicinity of true vertex locations and lower values elsewhere, as well as providing a criterion for when two vertex candidates are

resolved from each other. The definition of these functions is given and motivated in more detail elsewhere [17].

In the ZVMST algorithm, the efficient minimization of weights achieved by the minimum spanning tree is used to maximize the overall vertex function of selected vertex candidates. The ZVMST algorithm has two main stages: first a small number - typically between 1 and 5 - of 3D positions at which vertices are likely to be found is chosen on the basis of the vertex function. In the second phase tracks are assigned to these candidate vertex positions, using both the value of the Gaussian probability tube of each track at each of the selected space points and the vertex function value at these points.

To select the candidate vertex positions, the initial step is identical with that of ZVRES: for all possible two-track combinations in the input jet a vertex-fit is attempted, and combinations discarded that have a fit-χ^2 above a user-settable cut value (default: 10) or for which the vertex function at the resulting fit position is below 0.0001. The retained two-track combinations are used to set up a mathematical graph structure, in which each node corresponds to one of the tracks in the jet, and each edge corresponds to a successful vertex fit of the two tracks that it connects. Note that a connection is only made if the corresponding fit passes the cuts described above. As weight for the edge, the inverse of the vertex function at the vertex position obtained from the two-track fit is chosen.

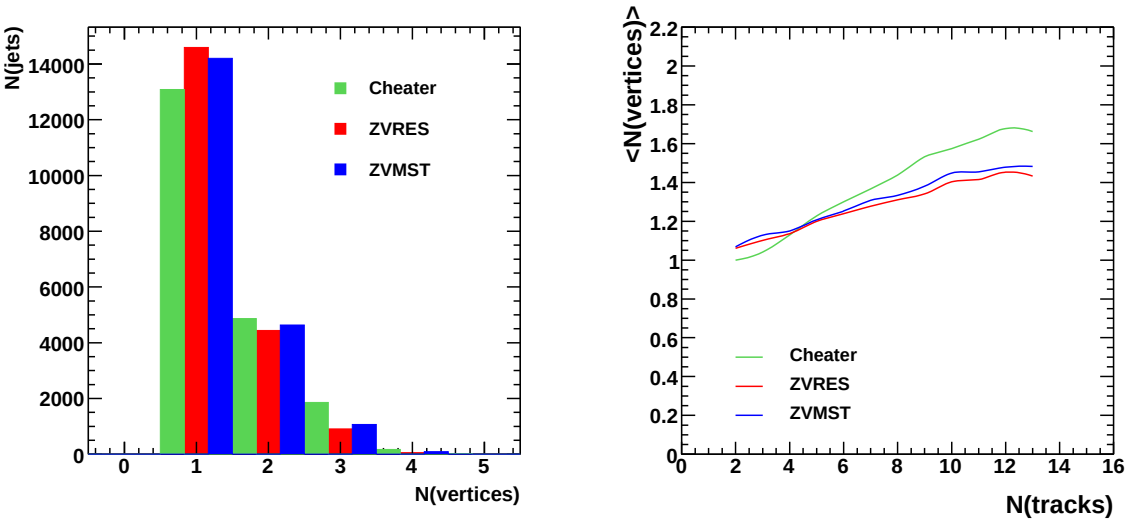
The graph is passed as input to the minimum spanning tree algorithm. This algorithm selects a set of at most $N-1$ edges for N input nodes (or less if the input graph contains unconnected nodes) in such a way that the overall weight is minimised. In this case, because of the choice of the weights, this minimisation corresponds to maximising the sum of the vertex function values for the selected two-track candidate vertices.

Often some of the $N-1$ selected candidates will correspond to the same physical vertex, especially for multi-prong vertices and the primary vertex. Therefore, sets of two-track candidates that correspond to one physical vertex need to be identified and only one optimized position derived for each set. The details of this stage of assigning tracks to vertices are described elsewhere [7].

5 Performance comparison of ZVMST and ZVRES at $\sqrt{s} = 91.2\,\mathrm{GeV}$

The performance of the new vertex finder ZVMST has been studied at a centre of mass energy of 91.2 GeV and compared to that of the standard ZVTOP_ZVRES algorithm. A cheater algorithm has been implemented which uses MC information to look up which tracks originate from the same space points, and passes these combinations through the same vertex fitter used for ZVRES and ZVMST. This cheater hence indicates the performance that could be achieved with perfect assignment of tracks to vertices.

Fig. 1 shows the inclusive vertex multiplicity and the vertex multiplicity as func-



(a) (b)

Figure 1: *Multiplicity of vertices found by the two topological vertex reconstruction algorithms ZVRES and ZVMST. The multiplicity of reconstructable vertices as found by the vertex cheater is shown for comparison. (Reconstructable vertices are all vertices that contain at least two tracks that are assigned to the same jet by the jet-finder). Shown are (a) the inclusive distribution and (b) the average vertex multiplicity as function of track multiplicity in the input jet.*

tion of the number of tracks corresponding to the vertex, for `ZVMST`, `ZVRES` and the cheater. Both vertex finding algorithms yield a smaller number of vertices than the cheater, with `ZVMST` finding a slightly larger number than `ZVRES`. In particular as the track multiplicity increases, the number of vertices falls short of the true MC vertex multiplicity by an increasing amount. At high multiplicity, `ZVMST` is closer to the MC truth, while this is the case for `ZVRES` at low multiplicity; the cross-over point is at a track multiplicity of about 4.

The performance of the track to vertex assignment for the two algorithms was studied and is discussed in detail elsewhere [7]. It yields similar performance for both algorithms, with some aspects being better for one, and others being better for the other.

Fig. 2 shows the flavour tag performance in terms of purity vs efficiency for a 91.2 GeV mixed sample with natural fractions of b-, c- and light jets. The result obtained from the `ZVMST` algorithm is compared to the performance of `ZVRES` and the cheater. The new `ZVMST` algorithm yields an improved c-tag purity over the full efficiency range, with the difference compared to `ZVRES` reaching values up to 5% at low

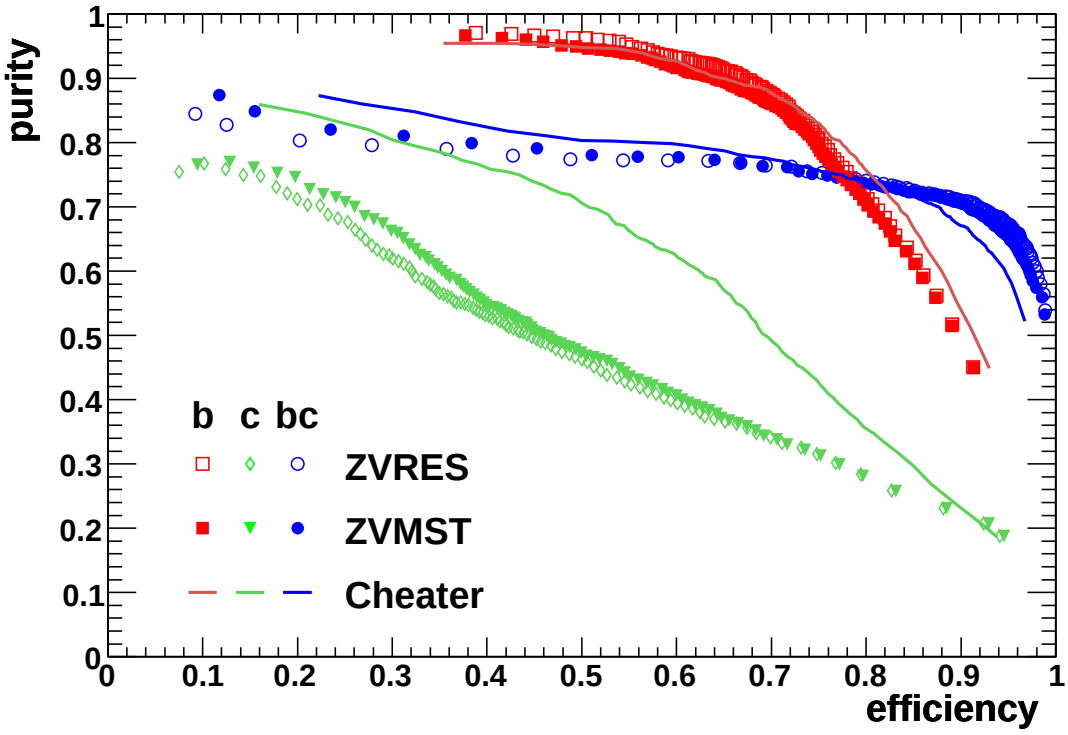


Figure 2: *Comparison of tagging performance at the Z-resonance obtained using the new ZVMST vertex finder compared to results obtained using ZVTOP's ZVRES algorithm and for a vertex cheater using MC information for track-to-vertex assignment. Tagging purity is shown as function of efficiency for b-jets and c-jets. Performance for c-jets assuming only b-background (labelled "bc") is also shown.*

efficiency. However, b-tag purity is degraded by up to 1.5% compared to `ZVRES`. This is consistent with the trend found during the code development that the c-tag is the most sensitive of the three tags whenever a change is made e.g. to track selection, track reconstruction, detector geometry etc. The comparison with the cheater performance shows that if improvements could be made to the track-to-vertex assignment this should directly result in an improvement in c-tag performance.

It should be noted that all three results are found using the flavour tag neural networks that were trained with the fast MC SGV at an earlier stage of code development. Since the time when the `ZVMST` study has been performed it was shown that flavour tag performance is sensitive to the networks used [27], and that a more realistic comparison would therefore require training dedicated networks to be used with each of the algorithms. This could also explain why in Fig. 2 the b-tag purity obtained from

the cheater is slightly worse than that obtained from the two realistic vertex finders.

6 Summary and conclusions

The LCFIVertex package, providing vertexing, flavour tagging and quark charge reconstruction, is an essential tool for the preparation of detector LoIs for the ILC, with generic applicability to high-precision vertex detectors. Additions to the core functionality of the code provided by the first release in 2007 include extensive diagnostics, technical improvements and work towards a default configuration of the code, as well as the new vertex finder `ZVMST`, based on a minimum spanning tree approach.

In a performance study of the `ZVMST` algorithm at $\sqrt{s} = 91.2\,\mathrm{GeV}$, the algorithm has been shown to be competitive with the leading algorithm at the ILC, `ZVTOP_ZVRES`. Vertex multiplicities from `ZVMST` are slightly closer to the reference values obtained from MC truth track combinations than is the case for `ZVRES`. Track-to-vertex assignment is similar for both algorithms, some aspects being improved for `ZVMST`, while others are degraded.

At $\sqrt{s} = 91.2\,\mathrm{GeV}$ flavour tag performance has been compared using the flavour tag networks obtained from the fast MC SGV, which have subsequently been shown not to be optimal for any of the algorithms if using GEANT4 based MC and realistic event reconstruction. Within these boundary conditions, `ZVMST` yields an up to 5% increase in c-tag purity and an up to 1.5% degradation in b-tag purity compared to `ZVTOP_ZVRES`. A further limitation of this study is the fact that the code parameters used are not yet optimized; though `ZVRES` parameters are set to the values obtained from an earlier GEANT3-based study and a preliminary parameter optimization at 91.2 GeV was performed for `ZVMST` [7].

References

[1] T. Behnke, S. Bertolucci, R. D. Heuer and R. Settles, *TESLA: The superconducting electron positron linear collider with an integrated X-ray laser laboratory.* Technical design report. Pt. 4: *A detector for TESLA*, ECFA-2001-209;

[2] ILC Global Design Effort and World Wide Study, J. Brau (Ed.) et al., *ILC Reference Design Report (RDR)*, SLAC-R-857 (2007), available from `http://www.linearcollider.org/rdr/`;

[3] D. Bailey et al. (LCFI Collaboration), *The LCFIVertex Package: vertexing, flavour tagging and vertex charge reconstruction for the design of an ILC vertex detector*, in preparation, to be submitted to Nucl. Instrum. Meth. A;

[4] available from `http://ilcsoft.desy.de/portal/software_packages/lcfivertex/`;

[5] `http://www.ilcild.org/`;

[6] `http://silicondetector.org/display/SiD/home`;

[7] S. Hillert, *ZVMST: a minimum spanning tree-based vertex finder*, LC-DET-2008-004, (2008) [arXiv:0811.3904];

[8] F. Gaede, T. Behnke, N. Graf and T. Johnson, *LCIO: A persistency framework for linear collider simulation studies*, in the *Proceedings of 2003 Conference for Computing in High-Energy and Nuclear Physics (CHEP 03), La Jolla, California, 24-28 Mar 2003, pp TUKT001*, [arXiv:physics/0306114];

[9] F. Gaede, *Marlin and LCCD - Software tools for the ILC* Nucl. Instrum. Meth. **A** 559 (2006) 177;

[10] Geant4 collaboration, S. Agostinelli et al., *GEANT4 - a simulation toolkit* Nucl. Instr. and Meth. **A** 506 (2003) 250; Geant4 collaboration, J. Allison et al., IEEE Trans. Nucl. Sci. **53** (2006) 1;

[11] P. Mora de Freitas, *Mokka, Main Guidelines And Future*, Prepared for *International Conference on Linear Colliders (LCWS 04)*, Paris, France, 19-24 Apr 2004;

[12] Browser for Detector Models in the MOKKA Geometry Database, http://www-flc.desy.de/ldcoptimization/tools/mokkamodels.php?\model=LDC01_05Sc

[13] O. Wendt, F. Gaede and T. Kramer, *Event reconstruction with MarlinReco at the ILC*, arXiv:physics/0702171;

[14] A. Raspereza, FullLDCTracking Manual available from http://ilcsoft.desy.de/MarlinReco/v00-10-04/doc/html/index.html;

[15] M. A. Thomson *Proceedings of LCWS06*, Bangalore, March 2006, [arxiv.org/physics/0607261];

[16] S. Hillert on behalf of the LCFI Collaboration, *The LCFIVertex package: vertex detector-based Reconstruction at the ILC*, Proceedings of the 2007 International Linear Collider Workshop LCWS 2007, DESY, Hamburg (2007), [arXiv:0708.2622];

[17] D. J. Jackson, *A topological vertex reconstruction algorithm for hadronic jets*, Nucl. Instrum. Meth. A **388** (1997) 247;

[18] J. Thom, "Search for B_s^0-$\bar{B}_s^0$ Oscillations with a Charge Dipole Technique at SLD ", PhD thesis, SLAC-R-585 (2002);

[19] R. Hawkings, *Vertex detector and flavour tagging studies for the TESLA linear collider*, LC-PHSM-2000-021-TESLA, available at http://www-flc.desy.de/lcnotes/notes/LC-PHSM-2000-021.ps.gz

[20] S. Xella-Hansen, D. Jackson, R. Hawkings, C. Damerell, *Flavour tagging studies for the TESLA linear collider*; LC-PHSM-2001-024 (2001);

[21] S. Xella-Hansen, M. Wing, D. Jackson, N. de Groot, C. Damerell *Update on flavour tagging studies for the Future Linear Collider using the Brahms simulation* LC-PHSM-2003-061 (2003);

[22] available from http://berggren.home.cern.ch/berggren/sgv.html;

[23] V. Adler, *Optimising of design parameters of the TESLA vertex detector and search for events with isolated leptons and large missing transverse momentum with the ZEUS-experiment (HERA II)*, DESY-THESIS-2006-012;

[24] K. Abe et al. (SLD Collaboration), *A measurement of R_b using a vertex mass tag*, Phys. Rev. Lett., **80** (1998) 660;

[25] S. Hillert on behalf of the LCFI Collaboration, *Quark charge measurement with an ILC vertex detector*, in: J. Hewett (Editor) Proceedings of the 2005 International Linear Collider Workshop LCWS, Stanford (2006) 341; available at `http://www.slac.stanford.edu/econf/C050318/papers/0313.PDF`;

[26] B. Jeffery, PhD Thesis, in preparation;

[27] R. Walsh, *Flavour tagging studies with the LCFIVertex package*, contribution to the International Linear Collider Workshop, LCWS08 and ILC08, Chicago (2008), to be published in the proceedings;

[28] R. Campana, E. Massaro, D. Gasparrini, S. Cutini, A. Tramacere, *A Minimal Spanning Tree algorithm for source detection in γ-ray images* Mon. Not. R. Astron. Soc. 383, (2008), 1166-1174;

[29] J. G. Siek, L.-Q. Lee, A. Lumsdaine, *Boost Graph Library, User Guide and Reference Manual*, C++ In-Depth Series, Addison Wesley Professional (2001).